DOCUMENT RESUME

ED 050 585 EM 008 945

| | |
|---|---|
| AUTHOR | Avner, R. A. |
| TITLE | TUTOR User's Memo; Execution of TUTOR Statements. |
| INSTITUTION | Illinois Univ., Urbana. Computer-Based Education Lab. |
| PUB DATE | 16 Apr 71 |
| NOTE | 37p.; See also TUTOR Manual, EM 008 943 and TUTOR Memo, EM 008 944 |
| EDRS PRICE | EDRS Price MF-$0.65 HC-$3.29 |
| DESCRIPTORS | *Computer Assisted Instruction, *Manuals, *Programing Languages |
| IDENTIFIERS | PLATO, Programmed Logic for Automated Teaching Operations, *TUTOR |

ABSTRACT

The information in this memo allows the advanced TUTOR author to predict the effect or outcome of any TUTOR statement in a given situation. Each TUTOR lesson consists of an ordered list of TUTOR statements. During use of the lesson, the computer is required to examine portions of this list and to perform operations specified by some or all of the TUTOR statements encountered during this examination. Seven situations are defined which specify all cases in which a statement in a particular part of a lesson might be examined. Statements which are executed during each situation are listed and details of unusual effects or interactions which can be expected are described. TUTOR commands are classified into eight functional annotated categories. A chart shows the TUTOR commands and summarizes the situation under which they are legal. General familiarity with the TUTOR language is assumed. (JY)

ED050585

U.S. DEPARTMENT OF HEALTH, EDUCATION & WELFARE
OFFICE OF EDUCATION
THIS DOCUMENT HAS BEEN REPRODUCED EXACTLY AS RECEIVED FROM THE PERSON OR ORGANIZATION ORIGINATING IT. POINTS OF VIEW OR OPINIONS STATED DO NOT NECESSARILY REPRESENT OFFICIAL OFFICE OF EDUCATION POSITION OR POLICY.

# TUTOR USER'S MEMO

## Execution of TUTOR Statements

**APRIL 16, 1971**

Suggestions for correction or improvement of this memo should be directed to R.A. Avner (364 ERL).

Computer-based Education Research Laboratory
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801

EM008 945

Execution of TUTOR Statements

Page

Introduction and Definitions.....1

Summary of Situations.....3

READIN Situation.....7

UNIT Situation.....13

Allocation of Storage Space for Student Responses.....14

SEARCH Situation.....16

Review Units.....20

ARROW Situation.....21

Effects of the -REPLOT- Key.....22

JUDGE Situation.....25

ANSWER Situation.....29

ENTRY Situation.....31

Appendices: A. Classification of Commands.....33

B. Legal Situations for Command Execution.....37

This memo is intended to provide the advanced TUTOR author with information which will allow him to predict the effect of any TUTOR statement in a given situation. General familiarity with the TUTOR language is assumed.

Each TUTOR lesson consists of an ordered list of TUTOR statements. During use of the lesson, the computer is required to examine portions of this list and to perform operations specified by some or all of the TUTOR statements encountered during this examination. For convenience, seven situations are defined which specify all cases in which a statement in a particular part of a lesson might be examined. This memo lists statements which are "executed" (i.e., used to perform their intended function) during each situation and describes details of unusual effects or interactions which can be expected.

Authors are directed to Appendix A for a list of 8 major categories of TUTOR commands. These categories are used extensively in the following pages. Appendix A also serves as a list of current TUTOR statements annotated with brief descriptions of their function.

Appendix B consists of an index of TUTOR commands and a summary of the situations under which they are legal.

## DEFINITIONS

The following special terms are used.

Statement - A TUTOR statement consists of a TUTOR command plus a tag which specifies the precise effect of that command in a given situation. Where a command is normally used with a blank tag (e.g., UPLOW, END, or RETRY), the terms "statement" and "command" may be used interchangeably.

Attached Unit - A Unit used as the result of execution of a JOIN, IJOIN, DO, or GOTO statement in a given Unit is said to be "Attached" to the given Unit. An Attached Unit may have Attached Units of its own.

Main Unit - A Unit containing statements which are currently being examined or executed but which is not an Attached Unit is said to be a Main Unit. There are three types of Main Units.

> Base Unit - The first Unit encountered by a student in a lesson is a Base Unit. Any Main Unit reached directly (i.e., by execution of a JUMP or a real or implied NEXT statement) from a Base Unit becomes a Base Unit. Any Main Unit for which a BASE command is executed becomes a Base Unit. The most recent Base Unit used by a student is his "current Base Unit" (or simply his "Base Unit").
>
> Branch Unit - A Main Unit reached by execution of a DATA, DATA1, HELP, HELP1, LAB, LAB1, or TERM statement is said to be a Branch Unit. Any Main Unit reached directly from a Branch Unit is also a Branch Unit. At completion of a sequence of Branch Units the student is returned to his current Base Unit.
>
> Review Unit - A Main Unit reached by execution of a BACK statement is said to be a Review Unit. Any Main Unit reached from a Review Unit is also a Review Unit. Exception: When the Main Unit from which Review began is again encountered as a Main Unit, Review ceases and the Unit is defined in its original manner (either a Base or Branch Unit).

Note that the above definitions of Unit types apply only at one specific time for a particular student. The same Unit may at different times be classed as an Attached, Base, Branch or Review Unit. Student responses made in the current Base Unit are "remembered" by the computer while the student is in Branch or Review Units. Upon return to the current Base Unit these responses are redisplayed.

SUMMARY OF SITUATIONS

1. READIN - During the time a lesson or group of lessons are being read into the computer for student use, ten TUTOR commands are used to provide special features for the lesson as a whole or to control the process of READIN. Commands used are:

| | | | | |
|---|---|---|---|---|
| CHAR | DEFN | PARTS | STOP | TITLE |
| COM | LDATA | START | STOP* | UPLOW |

2. UNIT - Certain statements which appear before the first ARROW statement in a Main Unit (including statements Attached prior to the first ARROW) are executed when the student arrives at the Unit. Statements whose execution is contingent upon the occurrance of this situation form what is known as a "Unit Contingency". If ARROW statements are present, storage space is prepared for possible student responses to every ARROW. LONG statements following ARROW statements are used in preparing this storage space. Commands used during the UNIT situation are:

| | | | | |
|---|---|---|---|---|
| ADD1 | ERASE | JOIN | PRESS | UDATA |
| [ARROW] | EXIT | JUMP | RANDP | [UNIT] |
| BASE | FAULT | LINE | RANDU | WHERE |
| CALC | FIND | LONG | REMOV | WRITC |
| CALCT | GOTO | LOOP | RPERM | WRITE |
| DO | IJOIN | MOVE | SHOW | WRIT* |
| [END] | INFO | PACK | SLIDE | WRUSC |
| [ENTRY] | INHIB | PAUSE | SUB1 | WRUSS |
| EQUIP | IPERM | PLOT | TIME | WRUS* |
| | | | | ZERO |

Note: Commands which will terminate the list of statements used for a UNIT contingency are enclosed in brackets.

3. SEARCH - Certain statements which appear within a Unit are used only at the time a student presses a particular key. At this time a search is made for a statement which allows the student to move to another Main Unit as a result of pressing the particular key. Statements used during the SEARCH situation (depending on which key was pressed) are:

| | | | | |
|---|---|---|---|---|
| [ARROW] | BACK | END | JOIN | NEXT |
| [ANS] | DATA | HELP | LAB | TERM |
| [ANSRU] | DATA1 | HELP1 | LAB1 | [UNIT] |

Note: Commands which delimit the area searched under some SEARCH situations are enclosed in brackets.

4. ARROW - Certain statements which occur between an ARROW statement and the next following Answer statement are executed whenever that ARROW is selected in a Base or Branch Unit. Statements used during the ARROW situation are:

| | | | | |
|---|---|---|---|---|
| ADD1 | EQUIP | LINE | RANDU | UDATA |
| [ANS] | ERASE | LOOP | REMOV | [UNIT] |
| [ANSRU] | EXIT | [MATCH] | RPERM | WHERE |
| [ANSV] | FAULT | MOVE | [SEN] | [WRGRU] |
| [ARROW] | FIND | [MUST] | [SENRU] | WRITC |
| CALC | GOTO | [ORDER] | SHOW | WRITE |
| CALCT | INFO | PACK | SLIDE | WRIT* |
| [CANT] | INHIB | PAUSE | [SPELL] | [WRONG] |
| [DIDDL] | IPERM | PLOT | [STORA] | WRUSC |
| DO | JOIN | PRESS | SUB1 | WRUSS |
| [END] | JUMP | RANDP | TIME | WRUS* |
| [ENTRY] | | | | ZERO |

Note: Commands which terminate the list of statements used for an ARROW contingency are enclosed in brackets.

5. JUDGE - Certain statements which appear between an ARROW statement and the next following ARROW statement (or the end of the Main Unit) are used during the time a student response at that ARROW is being matched against anticipated student responses. Commands used during the JUDGE situation are:

| | | | | |
|---|---|---|---|---|
| (ANS) | CLOSE | (MATCH) | PUTS | STORA |
| (ANSRU) | (DIDDL) | (MUST) | RETRY | STORE |
| (ANSV) | [END] | OPEN | (SEN) | [UNIT] |
| [ARROW] | JOIN | (ORDER) | (SENRU) | (WRGRU) |
| BUMP | LOAD | PUT | (SPELL) | (WRONG) |
| (CANT) | | | | |

Note: Commands which terminate the list of statements used for a JUDGE contingency are enclosed in brackets. Commands used during JUDGE to specify what student responses will be recognized as anticipated responses are enclosed in parentheses. Certain of the commands enclosed in parentheses can cause termination of the judging process if a student response matches the tag used with the command.

6. ANSWER - Certain statements which appear between an Answer statement and the next following Answer statement (or ARROW statement, or end of the Main Unit) are executed when a particular student response is recognized during the JUDGE situation. Commands used during the ANSWER situation are:

| | | | | |
|---|---|---|---|---|
| ADD1 | EQUIP | LOOP | RANDU | UDATA |
| [ANS] | ERASE | [MATCH] | REMOV | [UNIT] |
| [ANSRU] | EXIT | MOVE | [RETRY] | WHERE |
| [ANSV] | FAULT | [MUST] | RPERM | [WRGRU] |
| [ARROW] | FIND | NEXT | [SEN] | WRITC |
| CALC | GOTO | NODUP | [SENRU] | WRITE |
| CALCT | HELP | [ORDER] | SHOW | WRIT* |
| [CANT] | INFO | PACK | SLIDE | [WRONG] |
| [DIDDL] | IPERM | PAUSE | [SPELL] | WRUSC |
| DO | JOIN | PLOT | [STORA] | WRUSS |
| [END] | JUMP | PRESS | SUB1 | WRUS* |
| [ENTRY] | LINE | RANDP | TIME | ZERO |

Note: Commands which terminate the list of statements used for an ANSWER contingency are enclosed in brackets.

7. ENTRY - Certain statements which appear between an ENTRY statement and the end of the main unit (or the next ENTRY statement within the Main Unit) are executed when a key is pressed and the student is at the ENTRY statement. Commands used during the ENTRY situation are:

| | | | | |
|---|---|---|---|---|
| ADD1 | EXIT | LINE | RANDU | WHERE |
| CALC | FAULT | LOOP | REMOV | WRITC |
| CALCT | FIND | MOVE | RPERM | WRITE |
| DO | GOTO | PACK | SHOW | WRIT* |
| [END] | INFO | PAUSE | SLIDE | WRUSC |
| [ENTRY] | IPERM | PLOT | SUB1 | WRUSS |
| EQUIP | JOIN | PRESS | TIME | WRUS* |
| ERASE | JUMP | RANDP | [UNIT] | ZERO |

Note: Commands which terminate the list of statements used for an ENTRY contingency are enclosed in brackets.

**Brief Title of Situation-** READIN

**Description of Situation-** Author requests that one or more lessons currently on disk storage be made available for student use in core storage. (Material written by an author at a PLATO terminal is stored on disk storage. A translation of the material must be made and placed in core storage before the computer can use it for presentation at a PLATO terminal.) The computer begins making a statement-by-statement translation of the lesson material.

**Statements which are examined-** During READIN all statements in the requested lesson or lessons are examined in order, starting with the first statement of the first requested lesson and ending with the last statement of the last requested lesson. This procedure may be modified by the READIN contingent commands PARTS, START, STOP, and STOP* described below.

As READIN progresses, a list is compiled. This list consists of "illegal" commands, legal commands with illegal tags, UNIT commands with duplicate tags, and branching or attachment commands which specify names of Units not present in the lesson or lessons being read in. At the end of READIN the author is shown as much of this list as will fit onto one screen display. The lesson or lessons in core storage after READIN will behave as if statements with illegal commands or tags were not present. All references to Units with duplicate names will be handled as if the reference was to the Unit which was first encountered during READIN. References to Units which are not among those encountered during READIN will be ignored.

**Statements which are executed-** Statements used at this time either define portions of lessons to be translated or define special features to be made available during student use of the material. General effects of these commands are described below.

**PARTS** Specifies that only certain parts of the lesson are to be translated. Sections to be read in are delimited by START, STOP, and STOP* statements having an entry (or entries) as their tag which is identical to the entry (or entries) used as a tag for the PARTS statement.

**STOP** When a STOP command is encountered, translation is halted. Subsequent statements in disk storage are examined only for the purpose of identifying the end of a lesson or a "START" command. If either are encountered and other lessons or statements follow, translation resumes.

**START** Serves to resume translation after a STOP command has been executed and, when used with a non-blank tag, defines the beginning of a particular segment of the lesson which may be referred to by a PARTS statement.

**STOP*** Halts translation and examination of following statements in the same lesson. If other lessons remain to be translated, translation begins immediately with the next lesson on the READIN list.

CHAR — Specifies a special character designed by the author and referenced in the tag of a special display command ("PLOT"). Only a single CHAR statement for each special character need appear in all lessons which are read into core storage at the same time. If, however, the translation of the lesson containing the CHAR statement is later deleted from core storage, PLOT statements in other lessons read in at the same time but not deleted will cease to function for students. CHAR statements may appear anywhere in a lesson but are most conveniently located in Units by themselves which are not encountered by students.

COM — Specifies the name and amount of storage space which is available in common to all students using a lesson. The COM statement allows interpretation of references to common storage and must precede the first such reference in each lesson using COM storage. If common storage is used in a group of lessons read in together, a COM statement must appear in the first lesson read in and and in each other lesson which makes reference to common storage. Only one common storage space may be used by a lesson or group of lessons read into core storage with the same READIN. COM statements are most conveniently located at the beginning of the first Unit in a lesson.

**DEFN**

Allows the author to define or assign a name to special mathematical or logical expressions. The author's name may then be used in all following statements which permit use of mathematical expressions (including "STORA" which permits student use of expressions). When such statements are translated during READIN, the actual expression is substituted for the author's special name (or, in the case of "STORA" is admitted as a "legal" expression). All DEFN statements must be located together and physically precede the first reference to the special expression. See a complete description of the DEFN command in the memo "TUTOR Commands" for the option which allows redefinition of an expression name for a later section of the lesson.

**LDATA**

When an LDATA statement is encountered in a lesson during READIN, its tag is used to set up the list of types of student data which are to be recorded during student operation. If more than one LDATA statement is encountered in the lesson, the last one overrides all others. An LDATA statement must occur in <u>every</u> lesson for which student data is desired. LDATA statements are most conveniently located at the beginning of the first Unit in a lesson. During student operation the directions specified by an LDATA statement may be overridden temporarily in a Unit by use of the UDATA command.

TITLE

When a TITLE command is encountered during READIN of a lesson, its tag is used to give a name to the lesson in a temporary lesson directory. This directory is seen after sign-in by students who have not been assigned specific lessons (or by persons who sign in under the name "STUDENT"). If no TITLE statement appears in a lesson, only persons with records assigning them to the lesson may enter it as a student. If more than one TITLE statement is encountered during READIN, the last encountered in each lesson is used to title the lesson. TITLE statements are most conveniently located at the beginning of the first Unit in each lesson.

UPLOW

If an UPLOW command is encountered during READIN of a lesson, student-typed responses and standard written displays in the lesson may contain both Cyrillic and Latin characters in upper- and lower-case. If no UPLOW command is present in a lesson, student-typed responses and standard written displays will contain only upper-case ("capital") Latin characters. An UPLOW command must appear in <u>every</u> lesson in which both upper- and lower-case characters are to be used. The UPLOW command is most conveniently located at the beginning of the first Unit of each lesson.

Next Situation- At completion of READIN, students may begin student operation. The first situation encountered by a student is UNIT Initiation (which see) for the Unit specified in his student records. If no Unit is specified, the student begins operation in the first Unit in the lesson specified in his records (or selected by him if sign-in is under the name "STUDENT" or if no lesson is specified in his records).

Notes: See the TUTOR User's Memo "Using AUTHOR MODE" (page 9) for additional details on use of UPLOW.

Brief Title of Situation- UNIT (Initiate a Unit)

Description of Situation- A student arrives at a Base or Branch Unit during student operation.

Statements which are examined- All statements after the Unit statement are examined sequentially until either the end of the Main Unit or an ARROW, ENTRY, or PAUSE statement is encountered. The "end of the Main Unit" may be the last statement in an Attached Unit, the last statement in the Main Unit, or an END command in the Main Unit. END commands in Attached Units terminate only the Attached Unit.

Statements which are executed- The following statements are executed as they are encountered.

All Display Statements

All Computation Statements

All Information Storage and Access Statements

Sequencing and Control Statements (except HELP and NEXT)

All Attachment Statements

Next Situations-

If no ARROW statement is encountered in Unit, only the keys described for the SEARCH Situation and the key -REPLOT- are legal. The SEARCH Situation (which see) begins when a legal key is pressed. If -REPLOT- is pressed in a Unit with no ARROWs, and no ENTRY or PAUSE statements, the screen is erased and all statements except IJOIN will, if legal, be executed again.

If an ARROW statement is encountered, space is set aside for possible student responses (see Notes below) and the ARROW situation (which see) begins immediately.

If an ENTRY statement is encountered, the ENTRY situation (which see) begins as soon as a key is pressed.

If a PAUSE statement is encountered, UNIT Initiation continues as soon as any key is pressed.

Notes: Normally the student's screen is automatically erased upon beginning of Unit Initiation. If a JUMP is executed during UNIT Initiation the Main Unit becomes the Unit named in the tag of the JUMP statement and Unit Initiation continues in that Unit. No screen erase is performed on entry to the new Unit.

If a BASE command is encountered during Unit Initiation of a Branch Unit the Branch Main Unit is redefined to be the current Base Unit.

If a PRESS command for the student's station is encountered, it will not have an effect until after completion of Unit Initiation or until a PAUSE command is encountered.

If no ARROW statement is present, the message "PRESS-NEXT-" will automatically be displayed on line 19 (at the bottom of the student screen) after all statements have been executed or examined. This message may be inhibited by use of the statement "INHIB NEXT" in the Unit.

If key -REPLOT- is pressed after execution of all PAUSE statements, all Unit statements except IJOIN and PAUSE are redone as during the Unit Initiation. If a JUMP is performed as a result of such a REPLOT execution of Unit statements, the student screen will not be erased, storage space for student responses (see below) will not be reallocated, and prior responses and judgments will apply for ARROWs in the new Unit and be displayed there.

## ALLOCATION OF STORAGE SPACE FOR STUDENT RESPONSES

If an ARROW statement is encountered during the UNIT Initiation, storage space for possible student responses to all ARROWs in the Unit is immediately set aside and UNIT Initiation halts. Allocation of storage space is done as follows:

1. Four 8-character words of storage are set aside for each ARROW in the Unit.
2. If a LONG statement is present after an ARROW, additional storage may be set aside for that ARROW. One additional word of storage is set aside for each 8 characters (or fraction) above 32. A LONG with a tag of less than 32 will not reduce the amount of storage set aside for that ARROW. The maximum LONG tag permitted is 392 (49 words).

For example, in Unit TEST,

```
UNIT      TEST
WRITE     Type your
            Name
            Age
ARROW     2Ø8
LONG      35
ANS
ARRROW    3Ø8
LONG      3
ANS
```

nine words of storage would be allocated as soon as ARROW statements were encountered (four words for each of the two ARROWs plus one word for the 3 added characters specified by "LONG 35" after the first ARROW).

Total storage allocated for student responses in a single Base Unit plus that allocated for any one Branch (HELP, TERM, etc.) Unit reached from the Base Unit may not exceed 648 characters (81 words) of storage. Thus, a single Base Unit could have as many as 20 ARROW statements (using 4 X 20 = 80 words of storage), and a Base Unit using 77 words of storage could have any number of Branch Units containing no more than one ARROW each so long as no Branch Unit used a LONG with a tag greater than 32 (since 81-77=4, the minimum amount of storage required for a single ARROW).

Allocation of storage space is done only on initial entry to a Base Unit and is not altered upon subsequent entry (as upon return from a Branch Unit) or after a REPLOT. Storage space left over from the Base Unit allocation is erased and temporarily allocated to each new Branch Unit with ARROWs as that Branch Unit goes through UNIT Initiation. All storage space is erased each time a new Base Unit which contains ARROWs goes through UNIT Initiation. An error message will result during a UNIT Initiation in which storage requirements exceed the space available under the above limitations.

Responses stored in response storage space are used to provide complete ("filled in") displays when the -REPLOT- key is pressed in a Base or Branch Unit and when a student returns to his Base Unit after being in Review or Branch Units.

**Brief Title of Situation-**

SEARCH (Search for statement which allows movement to another Unit when a particular key is pressed).

**Description of Situation-**

Student presses a key, indicating his desire to move to another Main Unit.

**Statements which are examined-**

Depends on key or keys pressed. Within the area examined, JOIN statements are executed so that statements in Units attached by JOIN commands are examined.

**Statements which are executed-**

BRANCH KEYS

(Student presses key -DATA-, shifted -DATA-, -HELP-, shifted -HELP-, -LAB-, or shifted -LAB-). A search is begun for the appropriate statement (DATA, DATA1, HELP, HELP1, LAB, or LAB1, respectively). Statements between the current ARROW statement and the first following ANS, ANSRU, or ARROW statement are examined in order. If an appropriate statement is encountered and this statement can be executed (i.e., its tag names an available Unit), the branch is made immediately. If no statement is encountered or executed (or if the Unit contains no ARROW statements) a new search is begun at the statement following the UNIT command. Statements following the UNIT statement are examined sequentially for an appropriate Branch command which can be executed. If such a command is encountered the branch is made immediately and the new Main Unit is defined to be a Branch Unit. If an appropriate Branch command is not executed before an ARROW command, an END command, or the last statement in the Unit; the search is ended and the key press is ignored. No branches may be made during a Review (see Notes below).

BACK — (Student presses key -BACK-). Statements following the UNIT statement are examined sequentially for a BACK statement which can be executed. If such a statement is found, the branch is made immediately and the new Main Unit is defined to be a Review Unit for the student. If no such statement is found before the last statement in the Unit or an END command, the key will be ignored (Exception: in a Branch Unit the student will be returned to his current Base Unit if no BACK statement is executed).

TERM — (Student presses key -TERM-, types an expression, e.g., "OBFUSCATION", and presses key -NEXT-). All lessons in core (regardless of whether or not they were read in with the same READIN), are searched in the order in which they were read in. The search halts if a TERM statement with a tag which matches the first seven characters in the expression typed by the student is found. If a Unit containing such a TERM statement is found, the student is immediately branched there and that Unit is defined to be a Branch Unit for the student. If no such Unit is found, a message so stating (e.g., "OBFUSCATION CANNOT BE FOUND") is displayed at the bottom of the student screen.

NEXT — (Student presses key -NEXT- while in a Unit for which all required responses have been satisfied). Statements following the UNIT statement are examined in sequential order until an ARROW statement, an END statement, the last statement in the Unit, or an

executable NEXT statement is encountered. If a NEXT statement is encountered and executed, the student is immediately sent to the Unit named in the tag of the NEXT statement. The new Main Unit generally will have the same definition (Base, Branch, or Review) as the current Unit (Exception: if the Unit from which a Review began is encountered during a Review sequence, it will be so recognized and Review will cease). If neither a NEXT nor an END statement is encountered before an ARROW statement or the last statement in the Unit, the student will be sent to the Unit which immediately follows the current Main Unit. In this last case the branch will be made exactly as if a NEXT to the following Unit (an "Implied NEXT") had been executed. If an END command but no NEXT command is encountered in a Branch Unit, the student is returned to his Base Unit upon pressing key -NEXT-. The same situation (END command but no NEXT command) in a Base Unit results in termination of the lesson with an appropriate message being displayed on the student screen.

BACK1 (Student presses the shifted -BACK- key). This is, strictly speaking, not a SEARCH situation but is included here for completeness. If a student is in a Review Unit, he is immediately sent to the Main Unit (either a Base or Branch Unit) from which the Review began. If the student is in a Branch Unit, he is immediately returned to his current Base Unit. The key has no effect in Base Units.

Next Situation- UNIT (UNIT Initiation; see Notes below for the special case of Review).

Notes: The effects described above are summarized in the table below. Statements attached by JOIN statements in a searched area are also searched. NEXT and HELP statements have special effects when they follow an Answer statement. These special effects are described in the notes following the description of the ANSWER situation.

| Key Pressed | Command Sought | First Area Searched | Second Area Searched |
|---|---|---|---|
| -DATA- | DATA | ARROW | UNIT |
| shifted -DATA- | DATA1 | ARROW | UNIT |
| -HELP- | HELP | ARROW | UNIT |
| shifted -HELP- | HELP1 | ARROW | UNIT |
| -LAB- | LAB | ARROW | UNIT |
| shifted -LAB- | LAB1 | ARROW | UNIT |
| -BACK- | BACK | TOTAL UNIT | NONE |
| shifted -BACK- | NONE | NONE | NONE |
| -NEXT- | NEXT | UNIT | NONE |
| -TERM- (and phrase, -NEXT-) | TERM (with same phrase) | STORAGE | NONE |

Areas - ARROW - region between current ARROW statement and next following ANS, ANSRU, or ARROW statement.

UNIT - region between UNIT statement for current Main Unit and the next following ARROW or END statement, or the end of the Main Unit.

TOTAL UNIT - region between UNIT statement for current Main Unit and the next following END statement or the end of the Main Unit. Region after ARROW statement(s) is searched.

STORAGE - search includes all lessons currently read in for student use.

## REVIEW UNITS

A Unit entered as a result of execution of a BACK statement with the name of that Unit as its tag is a "Review Unit". During student operation Review Units may be identified by the presence of the word "REVIEW" at the bottom of the screen in place of the standard "PRESS-NEXT-" message.

The purpose of Review is to allow the student to see material complete with correct answers to all questions or problems. Only the following keys have any effect during Review:

-NEXT- moves the student forward through Review Units (Review ceases as soon as the Unit from which Review began is reached).

-BACK- moves the student back through Review Units only if BACK statements are present (Review ceases if the Unit from which Review began is reached).

-BACK1-(the shifted -BACK- key) moves the student immediately to the Unit from which the Review began and halts Review.

-REPLOT-gives the student a fresh screen image of the current Review Unit.

Correct answers for each ARROW are supplied only if the first Answer statement following the ARROW is an ANS or ANSRU statement with the desired correct answer as its tag. The judgement "OK" is automatically shown to the right of the response space.

The message "REVIEW" is not shown if "INHIB NEXT" occurs before the first ARROW in the Unit. The judgement "OK" will not be shown after "correct answers" for ARROWs which are followed by "INHIB OKNO" statements.

Other than the above-described uses of "ANS", "ANSRU" and "INHIB OKNO" statements (and the JOIN statement), no statements following the first ARROW are examined or executed during Review. ENTRY, IJOIN, and PAUSE statements are ignored in Review Units no matter where they occur in the Unit.

If a BASE command precedes the first ARROW in a Review Unit, Review immediately ceases and the Unit is redefined as the student's Base Unit. UNIT Initiation then is begun for this new Base Unit at the first statement in the Unit. Note that any statements preceding the BASE command in such a situation could thus be executed twice (once during Review but prior to the execution of BASE and once during UNIT Initiation of the new Base Unit). For this reason, if BASE is used it should be the first command after the UNIT statement.

| | |
|---|---|
| Brief Title of Situation- | ARROW (Prepare for a student response at a particular ARROW statement.) |
| Description of Situation- | An ARROW is selected by the student by use of key -ARROW-, by pressing key -NEXT- after being judged "OK" on a prior ARROW, or by pressing key -NEXT- without entering a response on a prior ARROW. After UNIT Initiation, the first ARROW in a Unit is automatically selected. |
| Statements which are examined- | All statements between the selected ARROW statement and the first following Answer statement. |
| Statements which are executed- | All Display Statements<br>All Computation Statements<br>All Information Storage and Access Statements<br>Sequencing and Control Statements<br>BASE, HELP, and NEXT)<br>Attachment Statements (except IJOIN) |
| Next Situation- | 1. JUDGE begins if the student types a response which is long enough to activate automatic judging or if judging is requested by pressing key -NEXT-. (Exception: if no response is typed and other ARROWs are present, pressing key -NEXT- selects another ARROW).<br>2. ARROW begins if the student selects another ARROW.<br>3. SEARCH begins if an appropriate Branch or Control key is pressed.<br>4. ENTRY begins if an ENTRY command is encountered among the examined commands.<br>5. ARROW continues if a PAUSE command is encountered and any key is pressed. |

Notes: After all commands have been examined and the first ANSWER command has been encountered, a small Arrow is plotted on the student screen in the position specified by the tag of the selected ARROW statement. The student response, when typed, appears on the screen to the right of this ARROW.

A student may select other ARROW statements by use of the -ARROW- key or by pressing key -NEXT- after his response to the current ARROW is judged "OK". Selection of a new ARROW causes erasing of the small arrow at the prior ARROW position, the last ARROW-contingent display, and the last ANSWER-contingent display of the prior ARROW. Note that if several Display statements are used to produce ARROW- or ANSWER-contingent displays, the only displays which will be erased will be the last ARROW-contingent display and the last ANSWER-contingent display. ARROW- or ANSWER-contingent displays produced by statements prior to the last Display statement in each situation will remain on the screen.

Movement to other ARROW statements is done in cyclical fashion. All ARROWs are selected in physical order and, if the -ARROW- key is pressed while at the very last ARROW statement, the first ARROW is again selected. Selection of a new ARROW may also be done by pressing key -NEXT- without entering a response.

Movement from a particular ARROW may be limited if the statement "INHIB MOVE" is positioned after the ARROW statement and before the first following Answer statement. This INHIB statement prevents selection of another ARROW until a response to the current ARROW is judged "OK". The INHIB command may also be used to inhibit display of the small arrow ("INHIB ARROW" is used between the ARROW statement and the first following Answer statement).

When an INHIB MOVE statement is used at an ARROW (or when only a single ARROW occurs in a Unit), pressing key -NEXT- initiates the JUDGE situation even if no response has been typed.

## EFFECTS OF THE -REPLOT- KEY

As soon as a message is displayed on a student TV screen at a PLATO III terminal, the message begins to fade from sight. Normally the time between first display and fading to illegibility is so long that the student requests a different display well before problems of legibility arise. However, in situations where the same display is to be watched for many minutes, a means of refreshing the screen is needed. The -REPLOT- key provides that means.

Refreshing of a screen is done by re-execution of certain statements in the Unit which was used to present the original display and by display of student responses which have been stored temporarily by the computer.

Return to a Base Unit after Review or Branching (or to a Branch Unit after Review) is a special case of REPLOT which follows the same rules. The following outline describes the effects of pressing key -REPLOT-:

I. In a Base or Branch Unit (or upon return to such a Unit after Review)

A. The screen is completely erased.

B. All statements prior to the first ARROW statement (except IJOIN, ENTRY, and PAUSE) are executed just as in UNIT Initiation. The fact that IJOIN is not executed during a REPLOT allows attachment of statements which are to be executed only upon initial entry to a Unit.

C. The first ARROW which has not been satisfied (answered with a response which has been judged "OK") is automatically selected and statements between this ARROW statement and the first following Answer statement are executed just as in the normal ARROW situation.

D. All prior student responses and judgements of these responses are displayed. Statements following Answer statements are not executed and any special markings (underlines, arrows, or slashes) of sentences resulting from prior judging are not shown. Student response storage is not erased or reallocated. (Exception: If the student signs in to continue a lesson, he is sent to the Unit which was his current Base Unit at the time he last signed out, i.e., at the end of the prior class session. All old responses made in the Base Unit are lost and the Unit behaves as if the -REPLOT- key had been pressed before any responses had been made.)

E. If a BASE statement occurs before the first ARROW statement, operation is exactly as UNIT Initiation for a Base Unit. No student responses are "remembered" and the first ARROW is automatically selected.

II. In a Review Unit. Operation is exactly like initial entry into a Review Unit.

A. All statements prior to the first ARROW statement (except IJOIN, ENTRY and PAUSE) are executed just as in Unit Initiation.

B. Correct answers are displayed for all ARROWs which have an ANS statement with non-blank tag as the first Answer statement following the ARROW statement. The judgment "OK" is shown after these answers (or after the ARROW position if no answer is displayed) for ARROWs which do not use "INHIB OKNO". No arrows are shown.

C. No statements following ARROW statements are executed.

| | |
|---|---|
| Brief Title of Situation- | JUDGE (Process a student response) |
| Description of Situation- | A student types a response in a Base or Branch Unit following ARROW. Judging is initiated by the student (when key -NEXT- is pressed) or automatically (when the number of keyboard codes produced equals a number preset by a LONG statement). If no response is typed and the Unit contains several ARROW statements the student can initiate JUDGE by pressing key -NEXT- only if the current ARROW has an "INHIB MOVE" statement. |
| Statements which are examined- | All statements between the current ARROW statement and the next ARROW, END, or UNIT statement. |
| Statements which are executed- | JOIN statements<br>All Answer statements which are matched<br>All Response Manipulation and Storage statements<br>RETRY statements |
| Next Situation- | ANSWER |

THE PROCESS OF JUDGING

At the moment judging is initiated the computer stores two copies of the student's response:

1. "Original copy" - this copy is never altered during the judging process and is used only to reset the Judge copy to its original state.
2. "Judge copy" - this copy may be altered during judging by execution of BUMP, CLOSE, MATCH, LOAD, PUT, or PUTS statements. It may be reset to its original state (by copying from the "Original copy") by execution of the RETRY command during sentence judging or "JUDGE RETRY" at completion of judging. The current contents of the Judge copy are stored in TUTOR variables during judging whenever a STORE statement is executed.

Judging is essentially a process of attempting to find the best match between the student's response (as represented by the "Judge" copy) and some anticipated response (as represented by the tags of various Answer classification statements). Matching may be on the basis of character-by-character comparison (provided by ANS, ANSRU, WRONG, or WRGRU statements) or on the basis of comparison of groups of characters separated by spaces, commas, periods, asterisks, or carriage returns. This last type of matching is known as sentence judging and is provided by use of SEN, MUST, CANT and DIDDL statements. Alternate responses in character-by-character judging are provided for simply by listing the responses as tags of separate ANS, ANSRU, WRONG, or WRGRU commands. Statements to be executed as a result of a character-by-character match between student response and expected response are positioned after the appropriate character matching statement. An expected "sentence response" is defined by a group of SEN, MUST, CANT and DIDDL statements. Alternate sentence responses are provided by separating the groups of SEN, MUST, CANT and DIDDL statements specifying each expected response with a RETRY command. Statements to be executed for a perfect sentence match follow the first SEN or MUST statement in the group of statements defining the expected sentence. All statements between the ARROW statement for the current response and the end of the Unit or the next following ARROW statement are examined and, if legal during judging, executed in the order in which they occur. JUDGE halts immediately and ANSWER begins when one of the following occurs:

1. A MATCH statement is encountered.
2. The Judge copy exactly matches an ANS, ANSRU, WRONG, or WRGRU statement or is within the tolerance of an ANSV statement.
3. A word in the Judge copy exactly matches a word in the tag of a CANT statement (or, if SPELL is in effect, when the response almost matches the tag of the CANT statement).
4. All words in the Judge copy match all words required in a group of SEN and MUST statements which specify a particular anticipated sentence.
5. If no sentence judging statements are present and SPELL is in effect, JUDGE halts if the Judge copy almost matches an ANS, ANSRU, WRONG, or WRGRU statement. The response is judged "SP" (possible spelling error) and statements following the SPELL statement are executed if legal in the ANSWER situation.

Since the Judge copy may be reset to its original state at several points during a sentence judging operation (by use of RETRY), the author may specify several alternative approaches to finding a match between the student's response and an anticipated response. Each of these alternative approaches is termed a "judging-cycle". Statements between the ARROW statement and the first RETRY statement, between subsequent RETRY statements, and between the last RETRY statement and the next ARROW statement or the end of the Unit comprise each Judge cycle. Because of certain interactions between statements legal during JUDGE, authors must follow these general rules within a judging cycle:

1. All statements used for character-by-character judging (ANS, ANSRU, WRONG or WRGRU) must precede the first statement used for sentence judging (CANT, DIDDL, MUST, or SEN) or they will be ignored.
2. All statements used to alter the judge copy (BUMP, CLOSE, MATCH, LOAD, PUT, or PUTS) should precede the first sentence judging statement in the judging cycle unless very special effects are desired.
3. Special options such as ORDER or SPELL must be initiated within each judging cycle (by execution of the relevant statements) prior to all Answer classification statements within that cycle for which the options are to apply. If the options are to apply to all expected responses in the judging cycle, they precede all Answer Classification statements in the cycle.
4. Sentence judging statements must precede each RETRY statement used.

The statements ANSV, MATCH, and STORA may be positioned anywhere within a judging cycle.

If an exact Match cannot be made during judging, a judgement is settled on in the following manner:

1. If the last judging cycle has no sentence judging statements: The last ANS, ANSRU, WRONG, or WRGRU statement with a blank tag in the judge cycle is chosen as the best match. This usage of a character-by-character judge statement with a blank tag provides what is known as a "universal answer". If no "universal answer" is present in the last judging cycle, the response is automatically judged "NO".

2. If the last judging cycle contains sentence judging statements:
   The best-match-to-date for all judging cycles is selected.
   A "best-match-to-date" judgment is decided at a given judging cycle if;
   a. Every MUST statement (and/or every required word in SEN statements) is matched but the response includes extra unrecognized words, or
   b. The total number of matched MUST statements (and/or required words in SEN statements) exceeds that for the prior best match.

Discrepancies between the "best match" and the student response are marked by underlining possible spelling errors, slashing wrong words, and showing missorderings and omissions by arrows. In the case of "best match" rather than "exact match", several Answer statements may be counted as having been matched. In such a case statements contingent on these matchings are performed in the order: SPELL, MUST (with blank tag), ORDER, and CANT (with blank tag). MUST with a blank tag is "matched" only if required words are missing. CANT with a blank tag is "matched" if an exact match was not possible.

Once in the ANSWER situation as a result of a match of any sort, the JUDGE command may be used to modify the judgement or to return to the JUDGE situation.

For a more detailed description of the effect of specific statements see their description in the Memo "TUTOR Commands". The same memo also contains a more detailed description of the interaction of statements during JUDGE.

| | |
|---|---|
| Brief Title of Situation- | ANSWER |
| Description of Situation- | During JUDGE one or more Answer classifier statements were found to provide an exact or close match to a student response. This match may be exact or qualified (as in a possible misspelling or misordering). JUDGE halts and ANSWER begins. |
| Statements which are examined- | All statements between each Answer statement which is "matched" and the next following Answer statement, ARROW statement, or the end of the Unit. |
| Statements which are executed- | All Display Statements<br>All Computation Statements<br>All Information Storage and Access Statements<br>JUDGE<br>Attachment Statements (except IJOIN)<br>Sequencing and Control Statements (except BASE and INHIB)<br>ENTRY |
| Next Situation- | ARROW - if response is judged "OK" and another ARROW is selected for answering.<br>UNIT - if all ARROWs are satisfied and the student decides to procede or if a HELP or JUMP statement is executed.<br>SEARCH - if a Branch key is pressed<br>JUDGE - if a "JUDGE REOPEN" or "JUDGE RETRY" statement is executed<br>ENTRY - if an ENTRY statement is executed |

Notes: If a student answer is partially or entirely erased after the ANSWER situation (or if "JUDGE IGNORE" is executed), the following occurs:

1. The last ANSWER-contingent display is erased,
2. The situation reverts to that described at the completion of the ARROW situation (i.e., wait for a student response on the current ARROW).

   ARROW contingent statements are <u>not</u> re-executed.

Execution of "JUDGE EXIT" has the same effect except that ANSWER-contingent displays are not erased.

See the description of JUDGE for information on what constitutes a "best match" to an Answer classifier statement. Under some circumstances (e.g., a misspelling when the SPELL statement is present) statements contingent on more than one Answer classifier may be executed for a single student response. See descriptions of individual Answer classifier commands for details.

At completion of all ANSWER-contingent operations an automatic judgment such as "OK", "NO", "SP", or "DP" is placed after the response. In sentence responses special markings may be provided to indicate wrong words (slash marks), possible misspellings (underlining), or, words out of order or missing (arrows). These special markings are not retained if key -REPLOT- is used after a judgment.

HELP and NEXT statements have a special effect when they follow a matched Answer classifier statement. HELP causes an immediate branch to the Unit specified in the tag of the statement. NEXT causes all keys except the key -NEXT- to be ignored and causes the student to be moved to the Unit named in the tag of the NEXT statement when this key is pressed.

If no ANSWER-contingent displays are produced and a response is judged "OK", the next ARROW in the Unit is automatically selected if any ARROWs remain unsatisfied. If ARROWs are answered out of order this automatic selection may result in reselection of an ARROW which has already been correctly answered.

If ANSWER-contingent displays are produced for a given response, the student remains at the current ARROW after judging (no matter what the judgment is).

| | |
|---|---|
| Brief Title of Situation- | ENTRY |
| Description of Situation- | An ENTRY command is executed. Following statements are examined (and executed if legal during ENTRY) each time a key is pressed by the student. |
| Statements which are examined- | All statements between the ENTRY command and the end of the Unit or the next following ENTRY command. |
| Statements which are executed- | Display Statements (except INHIB)<br>Attachment Statements (except IJOIN)<br>All Computation Statements<br>Sequencing and Control Statements (except BASE, INHIB, HELP, and NEXT)<br>FIND, INFO, MOVE, PACK and UDATA<br>ENTRY |
| Next Situation- | ENTRY - if another ENTRY command is executed (ENTRY situation then applies for statements following the new ENTRY command).<br>ENTRY - if no other ENTRY command is executed (Exception: ENTRY ends if a JUMP is made to a Unit which does not contain an ENTRY command).<br>UNIT - (without initialization procedures) if JUMP is executed to a Unit which contains no ENTRY command. |

Notes: A detailed description of operations during ENTRY is contained in the description of the ENTRY command in the memo "TUTOR Commands".

# APPENDIX A

## CLASSIFICATION OF COMMANDS

For brevity in describing the operation of TUTOR in different situations it is convenient to classify TUTOR statements into aribtrary groups of statements having roughly similar functions. Reference may then be made to all statements in a group (e.g., "Display Statements") rather than listing each and every member of the group (i.e., "WHERE", "WRITE", "SLIDE", "SHOW", etc.). Certain statements which have a variety of applications appear under several classifications.

I. Display Statements - Statements which provide or alter visual displays on the student screen or which control ancillary equipment used to present information to the student.

EQUIP - controls equipment at each student station
ERASE - erases portions of a TV screen display
FAULT - controls equipment for all student stations
INHIB - overrides automatic messages at author's option
LINE - draws lines on the TV screen
PLOT - displays special characters designed by the author
SHOW - displays information stored in the computer
SLIDE - superimposes photographic slides on the TV screen display
WHERE - positions displays on the TV screen
WRITC - displays one of a series of pre-stored messages (using Latin characters) contingent on the value of a number
WRIT* - same as WRITC
WRITE - displays a prestored message in Latin characters on the TV screen
WRUSC - same as WRITC but message is in Cyrillic characters
WRUS* - same as WRIT* but message is in Cyrillic characters
WRUSS - same as WRITE but message is in Cyrillic characters

II. Computation Statements - Statements which permit computational and logical operations on variables stored by the computer.

ADD1 - increments an integer by one
CALC - provides all common computational and logical operations
CALCT - same as CALC but integer arithmetic uses truncation rather than rounding
IPERM - sets up a list of integers for sampling without replacement
RANDU - provides random, uniformly distributed, numbers (sampled with replacement)
RANDP - provides random integers, sampled without replacement from a uniform distribution
REMOV - allows removal of integers from a sampling list
RPERM - allows reconstitution of a sampling list
SUB1 - decrements an integer by one
ZERO - sets an integer variable to zero

III. Attachment Statements - Statements which permit groups of statements to be used in a Unit without actually being copied into the Unit.

DO - permits repeated attachment of a group of statements
GOTO - permits attachment of a group of statements (statements in the same Unit but following GOTO are not executable)
IJOIN- permits attachment to be done if and only if the statement is encountered during UNIT Initiation.
JOIN - permits attachment of a group of statements (statements in the same Unit but following JOIN are executable)

IV. Student-Initiated Branching Statements - Statements which permit the student to gain access to another Main Unit by pressing a key. The author specifies the Unit which will be reached.

BACK - permits viewing a Unit with "correct answers" filled in by pressing key -BACK- (usually used to Review material seen previously)
DATA - permits branch by pressing key -DATA-
DATA1- permits branch by pressing shifted key -DATA-
HELP - permits branch by pressing key -HELP-
HELP1- permits branch by pressing shifted key -HELP-
LAB - permits branch by pressing key -LAB-
LAB1 - permits branch by pressing shifted key -LAB-
NEXT - permits movement to another Unit by pressing key -NEXT-
TERM - permits branch by typing a given term

V. Sequencing and Control Statements - Statements which permit the author to control the manner and sequencing of a presentation.

BASE - permits redefinition of student's current Base Unit during Branching or Review
END - defines the end of a Base or Branch sequence if in a Main Unit
HELP - forces an immediate branch after the student makes a particular response
INHIB- prevents access to correct answers, forces multiple-part responses to be made in a specified order
JUMP - forces an immediate movement to another Unit
LOOP - sets an upper limit on the number of statements which can be executed between key presses
NEXT - forces a branch (at a time decided by the student) after a particular response
PAUSE- delays execution of subsequent statements until a key is pressed
PRESS- permits specification of a key to be pressed automatically
TIME - causes key -NEXT- to be pressed automatically at a specified time after execution of the statement
UNIT - specifies a group of statements which are to be treated as a unit

VI. Information Storage and Access Statements - Statements which permit information to be stored or accessed during student operation (see also COM and LDATA under "Lesson Statements" and statements under "Student Response Manipulation and Storage Statements").

FIND - causes search through storage for specified information
INFO - causes specified information to be stored on magnetic tape
MOVE - allows transfer of information between storage locations
PACK - places specified information in storage
UDATA- specifies types of student information to be kept on magnetic tape for a given Unit in a lesson (overrides LDATA).

VII. Lesson Statements - Statements which provide special features for the lesson as a whole.

C - permits the author to insert comments for his own use which are not seen by students
CHAR - permits specification of specially designed characters for screen display by the PLOT command
COM - specifies common storage space which is accessable to all students using the lesson
DEFN - permits the author to rename a variable or define a mathematical function for a lesson
LDATA- specifies types of student information to be kept on magnetic tape for a given lesson (may be overridden by UDATA)
PARTS- specifies a section of the lesson to be made available for student use
START- used with PARTS to designate the beginning of a segment of a lesson
STOP - used with PARTS to designate the end of a lesson segment
STOP*- used with PARTS to designate the end of a lesson segment if no other segments follow
TITLE- provides a title for the lesson and permits use of the lesson by any student
UPLOW- permits use of upper case, lower case, and Cyrillic character sets

VIII. Student Response Processing Statements - Statements which permit recognition, manipulation, classification or other processing of student responses. This important classification is subdivided into several parts.

A. Answer Statements - Statements which allow classification of a response and permit execution of other statements contingent on such classification.

ANS - specifies a particular response to be judged "OK"
ANSV - specifies a particular numeric response (with a tolerance) to be judged "OK"
ANSRU- specifies a particular response (in Cyrillic characters) to be judged "OK"
CANT - specifies key words (and their synonyms) which are sufficient for rejection of a sentence
DIDDL- specifies words which may be ignored in evaluation of a sentence
MATCH- sets an index to a specified value upon recognition of a given response
MUST - specifies key words (and their synonyms) which are necessary for acceptance of a sentence

ORDER - checks for possible misordering of words in expected sentence responses
SEN - specifies a sentence (with acceptable synonyms) to be judged "OK"
SENRU - like SEN but sentence is in Cyrillic characters
SPELL - checks for possible misspelling of expected responses
STORA - checks a logical or mathematical expression for correct form
WRGRU - specifies an expected response (in Cyrillic characters) which is to be judged as wrong
WRONG - specifies an expected response which is to be judged as wrong

B. Response Manipulation and Storage Statements - Statements which store, alter or permit alteration of a copy of a student response during judging.

BUMP - eliminates specified characters from a student response
CLOSE - causes information spread over several storage locations to be collapsed and used as a student response
LOAD - causes stored information to be treated as a student response
MATCH - deletes a specified word from a student response
OPEN - spreads components of a student response over several storage locations to allow detailed processing
PUT - replaces specified characters with other characters on a one-to-one basis
PUTS - replaces a group of specified characters with another group of characters (original and replacement groups may differ in size)
STORA - evaluates and stores a numeric or logical expression provided by the student
STORE - stores a student response

C. General Response Processing Statements - Statements which permit student responses to be received and which provide general control over their processing.

ARROW - permits responses subject to standard processing techniques
ENTRY - permits responses subject to processing completely specified by the author
JUDGE - permits modification of a standard response classification
LONG - defines the length of a response for which judging is to be automatically initiated
NODUP - permits rejection of duplicate student responses
RETRY - removes effects of prior alterations of a student response (done during judging operations)

# APPENDIX B
## LEGAL SITUATIONS FOR COMMAND EXECUTION

| Command | READIN | UNIT | SEARCH | ARROW | JUDGE | ANSWER | ENTRY |
|---|---|---|---|---|---|---|---|
| 1. ADD1 | | X | | X | | X | X |
| 2. ANS | | | T | T | J | T | |
| 3. ANSRU | | | T | T | J | T | |
| 4. ANSV | | | | T | J | T | |
| 5. ARROW | | T | T | T | T | T | |
| 6. BACK | | | X | | | | |
| 7. BASE | | X | | | | | |
| 8. BUMP | | | | | X | | |
| 9. C | | | | | | | |
| 10. CALC | | X | | X | | X | X |
| 11. CALCT | | X | | X | | X | X |
| 12. CANT | | | | T | J | T | |
| 13. CHAR | X | | | | | | |
| 14. CLOSE | | | | | X | | |
| 15. COM | X | | | | | | |
| 16. DATA | | | X | | | | |
| 17. DATA1 | | | X | | | | |
| 18. DEFN | X | | | | | | |
| 19. DIDDL | | | | T | J | T | |
| 20. DO | | X | | X | | X | X |
| 21. END | | T | T | T | T | T | T |
| 22. ENTRY | | T | | T | | T | T |
| 23. EQUIP | | X | | X | | X | X |
| 24. ERASE | | X | | X | | X | X |
| 25. FAULT | | X | | X | | X | X |
| 26. FIND | | X | | X | | X | X |
| 27. GOTO | | X | | X | | X | X |
| 28. HELP | | | X | | | X | |
| 29. HELP1 | | | X | | | | |
| 30. IJOIN | | X | | | | | |
| 31. INFO | | X | | X | | X | X |
| 32. INHIB | | X | | X | | | |
| 33. IPERM | | X | | X | | X | X |
| 34. JOIN | | X | X | X | X | X | X |
| 35. JUDGE | | | | | | X | |
| 36. JUMP | | X | | X | | X | X |
| 37. LAB | | | X | | | | |
| 38. LAB1 | | | X | | | | |
| 39. LDATA | X | | | | | | |
| 40. LINE | | X | | X | | X | X |
| 41. LOAD | | | | | X | | |
| 42. LONG | | X | | | | | |
| 43. LOOP | | X | | X | | X | X |
| 44. MATCH | | | | T | J | T | |
| 45. MOVE | | X | | X | | X | X |

| Command | READIN | UNIT | SEARCH | ARROW | JUDGE | ANSWER | ENTRY |
|---|---|---|---|---|---|---|---|
| 46. MUST | | | | T | J | T | |
| 47. NEXT | | | X | | | X | |
| 48. NODUP | | | | | | X | |
| 49. OPEN | | | | | X | | |
| 50. ORDER | | | | T | J | T | |
| 51. PACK | | X | | X | | X | X |
| 52. PARTS | X | | | | | | |
| 53. PAUSE | | X | | X | | X | X |
| 54. PLOT | | X | | X | | X | X |
| 55. PRESS | | X | | X | | X | X |
| 56. PUT | | | | | X | | |
| 57. PUTS | | | | | X | | |
| 58. RANDP | | X | | X | | X | X |
| 59. RANDU | | X | | X | | X | X |
| 60. REMOV | | X | | X | | X | X |
| 61. RETRY | | | | | X | T | |
| 62. RPERM | | X | | X | | X | X |
| 63. SEN | | | | T | J | T | |
| 64. SENRU | | | | T | J | T | |
| 65. SHOW | | X | | X | | X | X |
| 66. SLIDE | | X | | X | | X | X |
| 67. SPELL | | | | T | J | T | |
| 68. START | X | | | | | | |
| 69. STOP | X | | | | | | |
| 70. STOP* | X | | | | | | |
| 71. STORA | | | | T | X | T | |
| 72. STORE | | | | | X | | |
| 73. SUB1 | | X | | X | | X | X |
| 74. TERM | | | X | | | | |
| 75. TIME | | X | | X | | X | X |
| 76. TITLE | X | | | | | | |
| 77. UDATA | | X | | X | | X | |
| 78. UNIT | | T | T | T | T | T | T |
| 79. UPLOW | X | | | | | | |
| 80. WHERE | | X | | X | | X | X |
| 81. WRGRU | | | | T | J | T | |
| 82. WRITC | | X | | X | | X | X |
| 83. WRITE | | X | | X | | X | X |
| 84. WRIT* | | X | | X | | X | X |
| 85. WRONG | | | | T | J | T | |
| 86. WRUSC | | X | | X | | X | X |
| 87. WRUSS | | X | | X | | X | X |
| 88. WRUS* | | X | | X | | X | X |
| 89. ZERO | | X | | X | | X | X |

SYMBOLS:

X - Command is executed during the situation
T - Command terminates the situation (but is not executed)
J - Command used during JUDGE for classification of student responses